Association for Information Systems
AIS Electronic Library (AISeL)

ECIS 2007 Proceedings

European Conference on Information Systems (ECIS)

2007

Towards Tool Support for Information Model Variant Management - A Design Science Approach

P. Delfmann
ERCIS, delfmann@ercis.de

R. Knackstedt
ERCIS, knackstedt@ercis.de

TOWARDS TOOL SUPPORT FOR INFORMATION MODEL VARIANT MANAGEMENT – A DESIGN SCIENCE APPROACH

Delfmann, Patrick, European Research Center for Information Systems (ERCIS), University of Münster, Leonardo-Campus 3, 48149 Münster, Germany, delfmann@ercis.de

Knackstedt, Ralf, European Research Center for Information Systems (ERCIS), University of Münster, Leonardo-Campus 3, 48149 Münster, Germany, knackstedt@ercis.de

Abstract

Information models are important instruments for deciders in enterprise engineering because they help structure the environment which is subject of a reorganisation task. The perspectives on the enterprise differ depending on engineers and tasks. This is why the number of models is increasing in enterprises as more different user groups exist. This can cause a lot of redundant work and inconsistencies. This paper reports on a design science approach to develop a modelling tool which is able to provide specific variants of information models, which are aligned to the different perspectives of user groups. Therefore the tool supports a set of configuration mechanisms, which are specialised on the definition of certain kinds of differences between the model variants. The supported differences concern model elements and concepts of the modelling languages including their representations and designations. The relevant kinds of differences were identified by explorative case studies and a survey of experts. The configuration mechanisms were conceptualised by data models in a search process divided in two steps. A prototype shows the feasibility of the developed concepts.

Keywords: Multi-perspective Information Modeling, Model Variant Management, Design Science, Tool Support, Meta Modeling.

1 INTRODUCTION

The multiplicity of components and their relationships makes the design of information systems a complex task. Information models are established instruments for coping with the complexity of enterprise engineering. Moreover, according to (Karimi 1988) and (Kottemann & Konsynski 1984), they have turned out to be the basis for successful information systems development.

Information models perform the task of representing problems in their current processing state (Newell & Simon 1972). In the line of problem solving a modeller aligns his actions with his individual constructed perception of reality. According to Luhmann (1990), subjective abstraction denotes an essential strategy for the reduction of complexity and is therefore determined pragmatically. Subjective abstraction increases the probability of problem-solving as it provides a representation of problems – which is characterised by reduced complexity according to the real-world – on which model users can discuss problems efficiently. Thereby, problems appear more solvable. Accordingly, the more a model complies with the user's subjective view on the problem, the better the quality (Darke & Shanks 1996).

This statement motivates approaches, which are aligned to special perspectives of model users or at least model user groups that support different views onto an integrated information model. Models which provide different model variants that only contain model elements which are relevant for a specific perspective are called multi-perspective models (Rosemann 1998). Exemplary approaches are the *Architecture of Integrated Information Systems* (ARIS) (Scheer 2000), the *Zachman Framework* (Zachman 1987), the *Open Systems Architecture for Computer Integrated Manufacturing (CIM-OSA)* (ESPRIT Consortium AMICE 1989), *MEMO* (Frank 1999), and *Viewpoints* (Finkelstein et al. 1992). A common characteristic of these approaches is that the realization of multiple perspectives is restricted to providing different modelling views which result in different model types.

In the following article we present a research project which analyses the demand of a more detailed approach to differ views on information models and develops a tool support for extended configuration mechanisms, which are not restricted to only build model views. Developing a tool support for multi-perspective information systems is a matter of the design science research. Hence, for the characterization of our research project the design science guidelines proposed by Hevner et al. (2004) can be applied:

1. *Design as an artefact*: In the focus of the development there is a methodical approach for the definition of perspective specific information model variants, of which each supports particular tasks of enterprise engineering.
2. *Problem relevance*: The relevance of the constituent elements of the methodical approach is shown by explorative case studies and a survey of 28 professionals of several branches in research and practice. For lack of space we abandon a theoretical derivation of the relevance of multi-perspective modelling and refer shortly on appropriate literature.
3. *Design evaluation*: The feasibility of the concept is evaluated by its realization as a software prototype. In future work we will evaluate the application of the tool in additional case studies and modelling experiments.
4. *Research contribution*: Multi-perspective modelling as presented in this paper includes the corporate (re)use of elements of a comprehensive enterprise model. By this the work contributes an approach to increase the efficiency of information modelling in practice. The feasibility of the concept is shown by the implementation of a software prototype. The economical effects should be subject of following empirical research. Furthermore, our approach constitutes a basis for the construction of multi-perspective reference models. Reference models based on our approach contain rules which formulate states of the relevance of modelling elements in specific contexts. Such reference models can be interpreted as theories of information modelling, because they formulate regards respectively presumptions about the adequate design of information models in different contexts of model use. Our approach proposes a methodical basis to explicit such states. The theoreti-

cal and empirical development of these states seems to be a promising perspective to deepen the body of knowledge about modelling for further modelling research.

5. *Research rigour*: In order to identify the requirements for our concept we followed several empirical research approaches. Through multiple explorative case studies we developed a classification of differences in which model variants used in enterprise engineering practice differ. By a survey with 28 professionals we could confirm this classification as relevant. On this basis we developed our approach. By a well-elaborated comparison with approaches in literature and with commercial tools we could show that the developed approach is original and progressive in a significant manner. To evaluate the feasibility of the approach we implemented our concepts by a prototypical software tool.
6. *Design as a search process*: Several circumstances add our research project the characteristic as a search process. The requirements analysis took place in two steps, the case study research and then a survey. Also the conceptualization was executed in two steps. The first conceptual data model for the identified configuration mechanisms was revised by a new data model that took aspects of the realization as a tool much more into account. The results of additional evaluation studies in the future will also be used to advance the concept and its tool support.
7. *Communication of research*: The results of the case studies and the survey, a data model as an overview of our conceptualization and a description of the prototype, are presented in this paper. For lack of space we present in the related work chapter only an excerpt of the comparison of our approach with others. The empirical data was conveniently archived.

The tool based approach for multi-perspective information models was developed with the help of empirical studies which explore requirements of the practice for the definition of perspective-specific model variants. Firstly, experiences form modelling projects in practice serve as a basis for the categorization of different model variant building forms (section 2). Secondly, the relevance of these variant building forms as well as the need for appropriate tool support is confirmed by a survey (section 3). Variant building forms that were identified as relevant are conceptualised by data models in a search process divided in two steps (section 4). The concept is evaluated by its realisation as a software prototype (section 5). The realised tool differs significantly from other research approaches and is beyond the functionality of actual versions of commercial modelling suites (section 6). Finally the following research will extend the evaluation of the tool application in different ways (section 7).

2 INFORMATION MODEL VARIANT BUILDING REQUIREMENTS

In multiple modelling projects that – among others – were realised at the *DeTe Immobilien GmbH* (Becker, Kugeler & Rosemann 2003), in *public administrations of the German federal state of North Rhine-Westphalia* (Becker et al. 2006a), at *Bayer Business Services GmbH* (Becker et al. 2006b), in the *German Federal Armed Forces* as well as in association with the *itemis GmbH & Co. KG,* different user groups mentioned the necessity of adapting models to their distinct requirements. The adaptations referred to were related to the modelling language, the model content as well as the graphical format of the models. Different requirements of user groups that lead to model adaptations are referred to as user *perspectives* (Rosemann 1998).

The following adaptation example illustrates which adaptation forms were requested in the projects in order to create model variants. An exemplary adaptation process is described which summarises the different aspects of model adaptation (cf. figure 1). In the example, a simplified business process of invoice auditing is shown and represented as an Event-Driven Process Chain (EPC) (Scheer, 2000). It is suited for retailing companies that run different types of businesses (warehousing, third-party-deal, and central settlement).

In a first step, the process model is adapted to requirements of a branch office that performs only two types of business (warehousing & third-party-deal). As a consequence, process branches that only serve the transaction type of central settlement are erased. Furthermore, the process model is adapted to requirements of practitioners that could be confused by the special syntax of the EPC that claims a strict alternating of functions and events. Semantically, not all events are necessary, since they just mark the successful termination of a function and do not provide further information concerning the following process flow. These so-called *trivial* Events are erased as well (adaptation step 1→2, cf. elements shaded grey in 1).

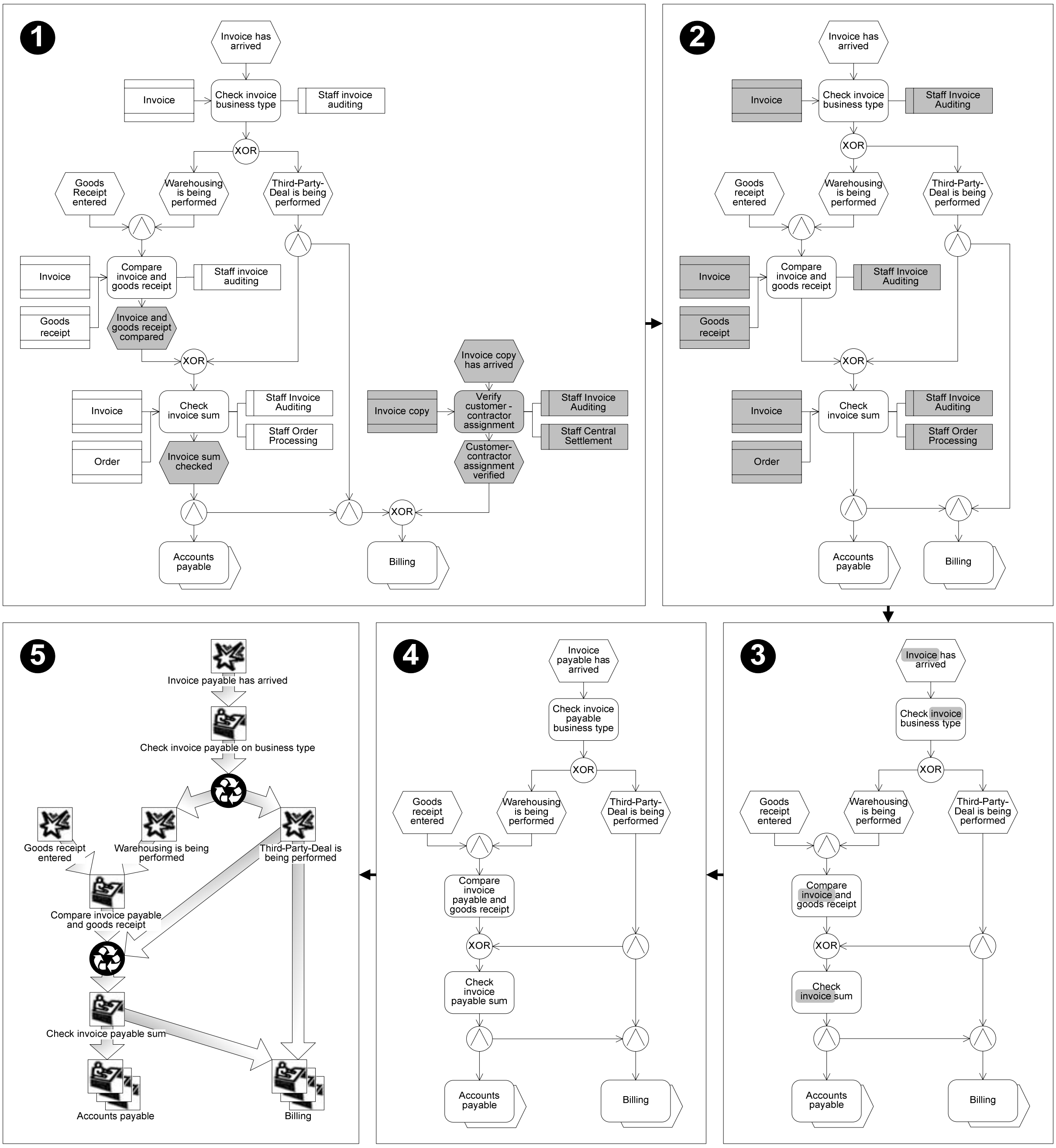

Figure 1: Exemplary adaptation process (Becker, Delfmann & Knackstedt 2007)

In a second configuration step, the process model is adapted in order to produce an overview of the process that reduces the model to an EPC without annotated resources. This is e.g. relevant for consolidation meetings within distributed modelling environments. Thus, the resource types “data cluster“ and “job“ are erased (adaptation step 2→3, cf. elements shaded grey in 2).

The third adaptation step considers different naming conventions. If the process model is to be provided to employees of distribution, the denotation “invoice“ may be misinterpreted as “invoice receivable“, whereas “invoice payable“ is meant here. Thus, each occurrence of “invoice“ is exchanged by “invoice payable“ (adaptation step 3→4, cf. text shaded grey in 3).

The graphical editing of information models can be used to overcome acceptance problems concerning formal models. Pictograms instead of polygons representing model elements are more likely to suit non-skilled model users (adaptation step 4→5).

A conventional adaptation of information models to the requirements of different user groups – as shown in the example – requires the creation of separate models. This causes increased construction and maintenance costs. In order to prevent these costs it is reasonable to make use of configurable information models. Configurable models contain rules that specify how the model has to be adapted according to user groups’ requirements. Model elements that are not relevant are hidden rather than erased. This makes the construction of separate models dispensable. Based on the requirements five categories of adaptation mechanisms, so called configuration mechanisms, are introduced (Becker, Delfmann & Knackstedt 2007):

- *Element Type Selection* considers the necessity to provide modelling language variants with different expressive power for different user groups. E.g., practitioners prefer process models that are easy to read. This can be achieved by e.g. fading out resource types that are annotated to process functions (cf. adaptation step 2→3).
- *Element selection*: Element selections allow for selecting single instances of model element types, e.g. a single process model function “Verify customer-contractor assignment” or trivial events (cf. adaptation step 1→2). Based upon this selection several elements which are not relevant for the certain user groups are faded-out.
- *Synonym Management*: This mechanism considers that it can be necessary to exchange the label of model elements in dependency of different user groups (cf. adaptation step 3→4).
- *Representation variation*: The variation of representation aspects allows the assignment of different representational forms to model elements. Hereby it is made possible to exchange model element symbols (cf. adaptation step 4→5) depending on the current perspective.
- A further model variant building form not shown in the adaptation example for clarity reasons is *Model Type Selection.* It allows for providing only modelling languages and their according model types to users that are relevant for them. E.g., employees who use process models as guidelines for their everyday work, do not need to be provided with data models describing data base structures.

3 SURVEY OF VARIANT BUILDING RELEVANCE AND EFFORT

The classification of configuration mechanisms derived from practical experiences was used to set up a questionnaire-based survey (for details cf. Delfmann & Knackstedt 2007). The aim of the survey was to analyse:

- if model variant management and especially the introduced configuration mechanisms are regarded as relevant by practitioners and researchers, and
- how time-consuming and costly the task of model variant management is estimated against the background of available tools supporting configurable information models.

177 researchers that have publicised on relevant topics of leading information modelling conferences were invited to participate in the survey. Furthermore, 170 members of the networking portal *openbc.com*, which had mentioned appropriate fields of interest, were asked if they would like to participate in the survey. After they had evinced interest in the survey, login information for the survey was send to 28 portal users. In total, 14 researchers and 14 members of the networking portal have participated in the survey and filled out the question form which was available over the survey portal *2ask*. The participants’ professions were distributed on different vocational branches in research and practice – according to the mixed acquisition strategy (14 times university, 1 time other research organization, 4 times industrial enterprise, 7 times consulting, 1 time other service enterprise, 1 time other). Within this survey, we do not claim representativeness regarding a certain basic population.

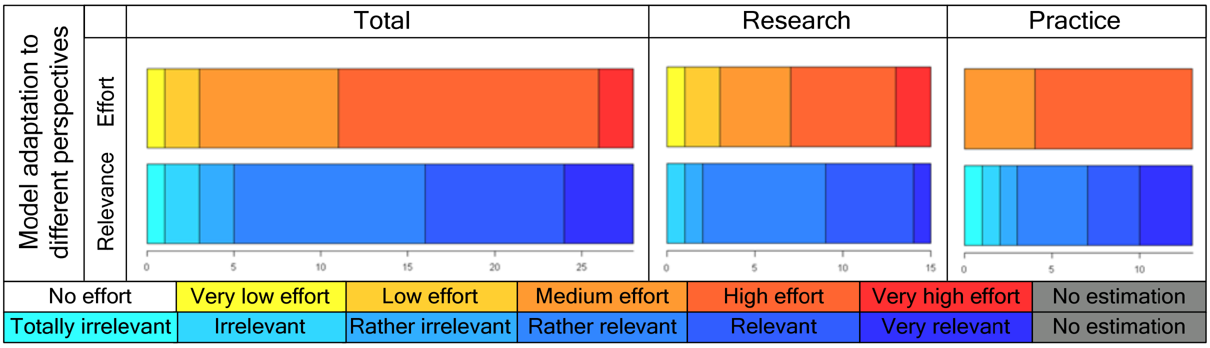

Figure 2: Effort and relevance of model adaptations to different perspectives

The results of the survey showed that user perspectives are predominantly considered as relevant triggers for the creation of model variants. The effort that is necessary to create and maintain appropriate perspective specific model variants is rated high in most cases. The estimations of participants from research and practice correspond (cf. figure 2).

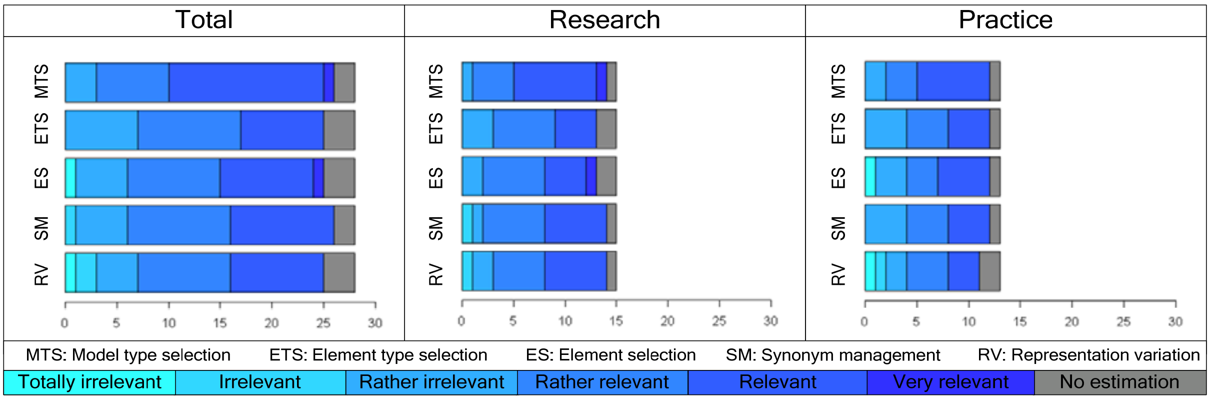

Figure 3: Relevance of variant building forms

Any of the different model variant building forms is rated as relevant for model variant management in most cases. The evaluations of the participants from practice and research also correspond to this key issue (cf. figure 3).

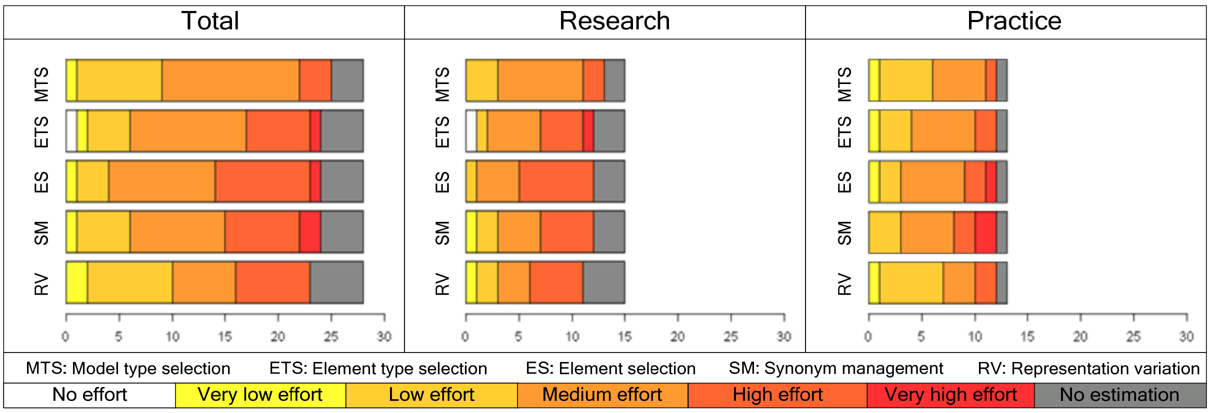

Figure 4: Effort estimation

On the one hand, the participants confirm the importance of the different variant building forms. On the other hand, they estimate a relatively high effort that has to be spent in order to perform variant building with one of the variant building forms. The estimation considers tool support that is available at present. The evaluations of research and practice are not very different here as well (cf. figure 4).

4 CONCEPTUALIZATION OF A CONFIGURATIVE INFORMATION MODELLING TOOL

The results of the survey have shown that an appropriate support of model variant management by modelling tools is desirable but not available. Thus, it was decided to develop a modelling tool ("*in-dapta*") that is able to provide appropriate model variant management based on the model variant building forms identified and confirmed as relevant.

As a methodical basis, we use the *configurative information modelling* approach by (Becker, Delfmann & Knackstedt 2007) that realises the identified model variant building forms providing corresponding configuration mechanisms (cf. section 2). These configuration mechanisms allow for annotating rules to information model elements that define if the element is relevant for a certain perspective or not. Running the configuration mechanisms depending on a perspective provides corresponding perspective-specific models, in which the elements that are marked as non-relevant are faded-out.

In a first step, the conceptual specification of configuration mechanisms was developed using different language specification layers in order to be able to modify the used modelling languages dynamically. For this purpose, a three-layer meta model system containing meta models and meta meta models was introduced (a detailed conceptualization can be found in Becker, Delfmann & Knackstedt 2007).

In order to implement the concepts of the first developing step, we consolidate the three-layer meta model system by constructing one single data model in a second step, which is the basis of the configurative information modelling tool. Thus, the data model contains both language specification aspects and model content aspects. Within the data base that is represented by this model, we store all information representing perspectives, modelling languages, models, model elements, model element relations as well as the relationships of all these issues. Hereby, we are able to select any modelling aspect depending on a certain perspective in order to show or hide it. The data model that is represented as Entity-Relationship Model (Chen 1976) is shown in figure 5.

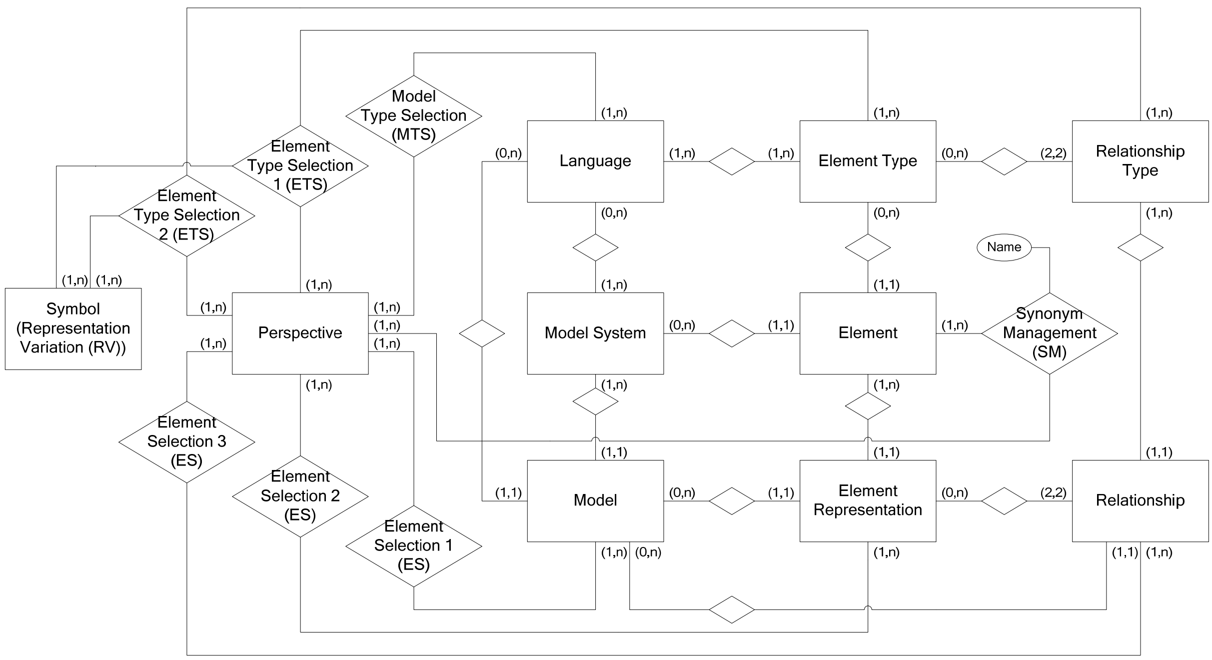

Figure 5: Configurative information modelling tool data model

Languages consist of *element types* that – in turn – are related to each other by *relationship types*. A *model system* contains different interrelated *models* that are the result of modelling with different modelling *languages*. Each model *element* belonging to a defined *type* is defined once but can be re-used in different models as an *element representation*. These *element representations* are interrelated by *relationships* that each belong to a certain *relationship type*.

Depending on a *perspective*, *languages*, *element types*, *relationship types*, *models*, *model element representations*, and *relationships* can be selected. The respective selection determines that one of the mentioned issues is shown or hidden. This way, the configuration mechanisms of *model type selection*, *element type selection*, and *element selection* are realised. Since *models* and *relationships* can be conceptually interpreted as elements as well, the selections of *models*, *element representations* and *relationships* are subsumed under *element selection (1-3)*. Analogously, the selections of *element types* and *relationship types* are subsumed under *element type selection (1-2)*.

Since *model systems* represent the whole model base and *elements* only represent the definitions of model *element representations*, it does not make sense to assign them to *perspectives* (e.g. hiding a *model system* would hide all models available; hiding a model *element* definition would hide every corresponding representation in different models). Besides these selections, *element types* and *relationship types* are assigned to *symbols* that can be exchanged according to the current *perspective*. Hereby, *representation variation* is realised. Furthermore, in order to enable *synonym management*, model *element names* are defined depending on the *perspective*.

5 TOOL APPLICATION

According to the data model introduced in section 4, the modelling tool *indapta* provides a language specification environment, a common modelling environment as well as a perspective management environment that enables a configuration of the constructed models. *indapta* uses the graphical drawing environment of Microsoft Visio as presentation layer. In the following, we show how the configuration steps 1→2, and 2→3 (cf. section 2) are specified in the perspective management environment.

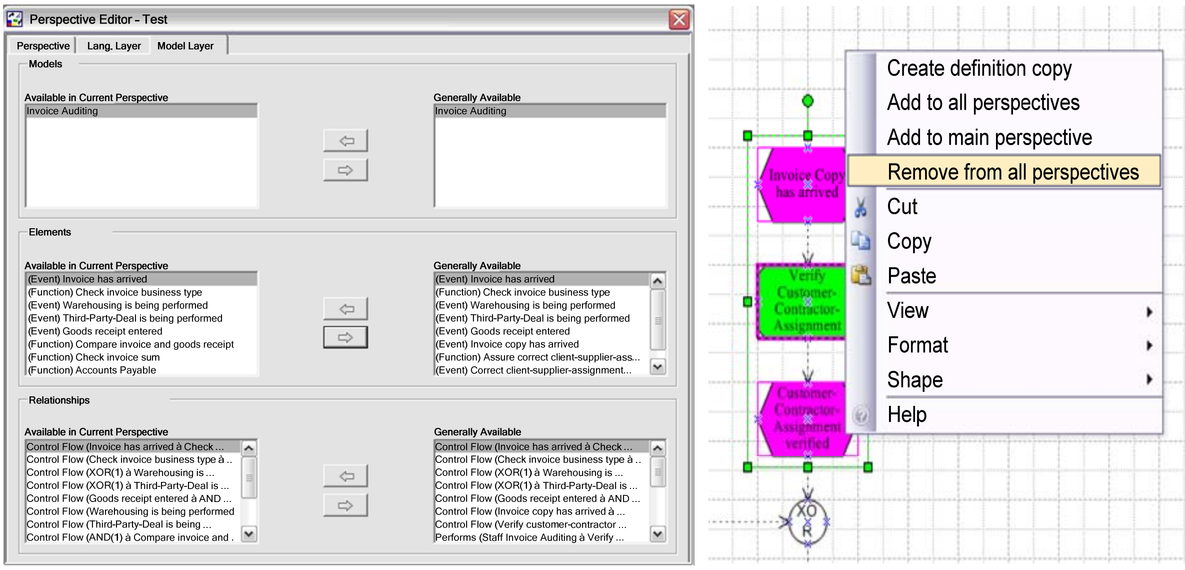

Figure 6: Perspective management environment (model layer)

In order to realise configuration step 1→2, distinct model elements have to be faded out. In the perspective management environment, they are excluded from the target perspective “Test” by not assigning them to the model elements available in the perspective (e.g., the model element “Invoice copy has arrived” is excluded). The same applies for the relationships that shall be excluded from the target perspective (cf. figure 6). For usability reasons, assignments or exclusions of elements to or from perspectives can not only be done in the perspective management environment. This can also be done graphically by modelling within the modelling environment that is switched to the according perspective.

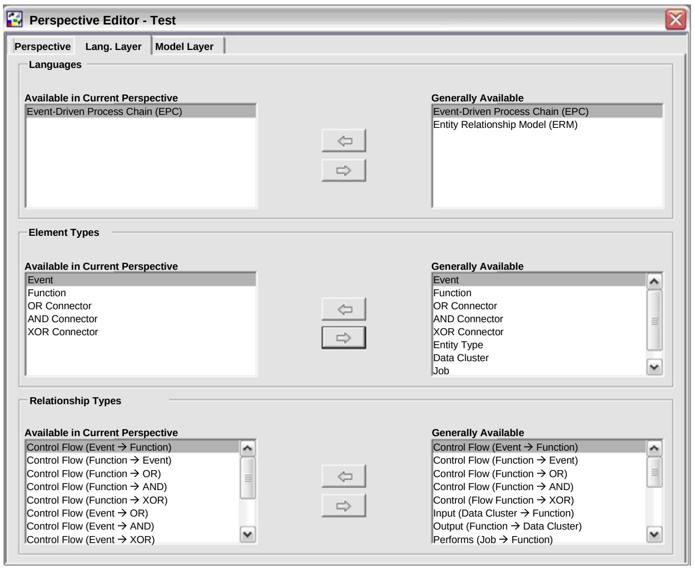

Figure 7: Perspective management environment (language layer)

In configuration step 2→3, all model elements that belong to the element types “data cluster” or “job” have to be hidden. This is realised by excluding the corresponding element types from the perspective “Test” within the perspective management environment (cf. figure 7). In turn, the relationship types that are assigned to these element types are excluded as well (e.g. “input”, “output” or “performs”).

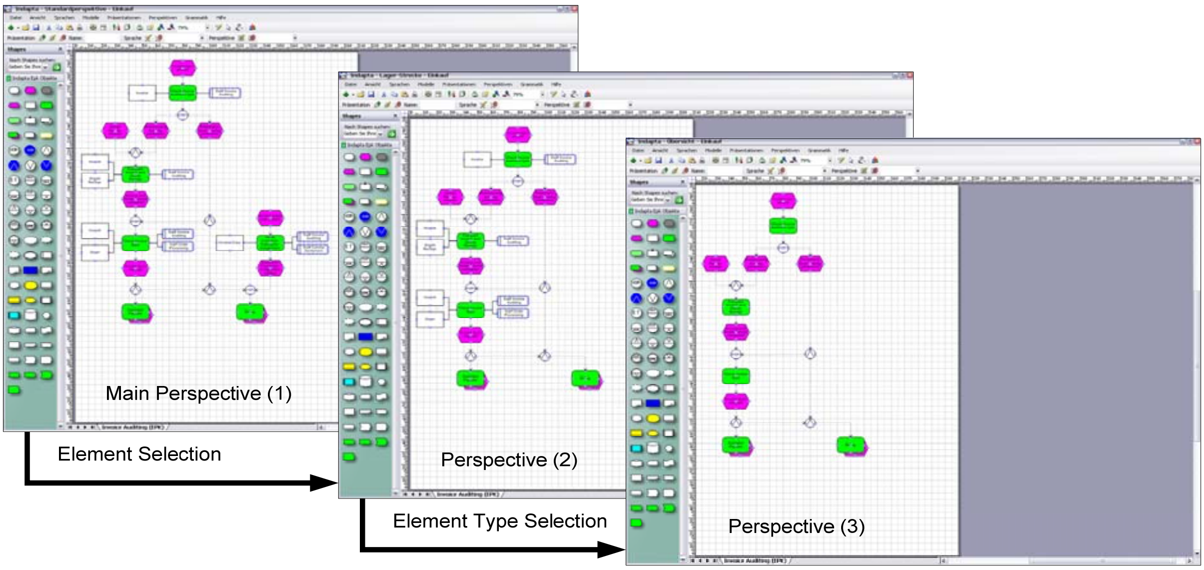

Figure 8: Modeling environment and configuration of different perspectives in indapta

The results of the exemplary configurations performed in steps 1→2 and 2→3 are illustrated in figure 8. It shows the modelling environment of *indapta* that is switched to the different perspectives. They are generated from the specifications made in the perspective management environment, and they represent perspective-dependent views on the underlying model database, whose data model we have introduced in section 4.

6 RELATED WORK

On the one hand, the maintenance, management and generation of information model variants are discussed in several articles that focus on an appropriate methodical conceptualization. On the other hand, modelling tools are available that claim to support model variant management. In the following, the reasons why the use of the configurative information modelling approach by (Becker, Delfmann & Knackstedt 2007) was decided to be used and why the decision was made to implement a new tool, rather than to reuse an existing one will be discussed.

(Soffer, Golany & Dori 2003) propose configurable reference models in order to customise enterprise systems. They use configurable, so-called Object-Process Diagrams that integrate process flows and data objects used within an enterprise system. The configuration of these diagrams is performed by interpreting attributes that define the relation of diagram objects to different application scenarios. During enterprise systems customizing, users have to specify their application context. Based on this, the attributes are interpreted, and the models are modified accordingly.

(Rosemann & van der Aalst 2007) propose a configurable reference modelling language that is based on EPCs. The approach differs from that of Soffer, Golany & Dori as configurations are less predefined. It is based on patterns in process models that describe semantic dependencies of model elements. E.g., a manual model configuration step that erases a process branch is followed by a hint to erase another process branch that is semantically related to the prior one. Similarly to Soffer, Golany & Dori, the authors point out the necessity to connect model elements to the according enterprise systems functions in order to perform a model and enterprise systems configuration concurrently.

In comparison to the approaches of Soffer, Golany & Dori and Rosemann & van der Aalst, the approach introduced by (Becker, Delfmann & Knackstedt 2007) which is used in this article is different as several configuration mechanisms are provided that have different influences on the models. The approach provides a set of configuration mechanisms that are able to format modelling languages, models and model sections as well as model elements in order to fit the model base to specific requirements. Furthermore, the approach is not restricted to conceptual configurations of information models but allows also configurations of the graphical representation of models as well as the management of language-internal synonyms.

The configuration strategy of each of the approaches is similar. Model variants for different application scenarios are integrated into one model and are predefined. The model variant that is considered the best for a specific application scenario can be selected. Therefore, these approaches provide a useful means of reducing the efforts for model adaptation, since the adaptation of models to different purposes is supported methodically.

In this paper, we follow the approach of (Becker, Delfmann & Knackstedt 2007) because it provides a comparatively extensive and detailed set of configuration mechanisms. Furthermore, it complies with the requirements of model variant management that have been identified in practice and confirmed by the survey, so that it is the most appropriate approach for the purpose of this article.

Based on the results of the survey, in which all identified configuration mechanisms were relevantly classified, different modelling tools were analyzed regarding their configuration support. Especially *meta modelling tools* were analysed, because the meta modelling environments provided by these tools promised to be appropriate specification environments for configuration mechanisms. Besides this group of tools, the product of the market leader in modelling tools, the *ARIS Toolset*, was included into the examination. Generally only those tools were evaluated which were available as trial version or which were available due to already existing university licenses. The analysis revealed that none of the tools provides an appropriate model variant support as follows (details of the tool analysis can be found in Delfmann & Knackstedt 2007):

- *ARIS*: Basically, the ARIS Toolset is not designed for configuration based model variant support. The script language which comes with ARIS provides rudimentary variations of models as well as representation variation of model element symbols. Furthermore, the assignment of variant building rules is possible. However, the toolset does not support querying perspectives with the objective of variant building. Basic configurations are provided by the so called method filter that manages the supply of model types and element types user-specifically. In ARIS, graphical variations are provided through different model types, which can be transformed into one another.

- *casewise*: This modelling tool provides the specification of variant building rules. However, the script language functionality is not sufficient in order to provide an automatic administration of such rules. The same applies for the query of perspectives as well as the specification of configuration mechanisms. Model type selections are provided by a filter. The modification of symbols is possible, however not by an automated configuration mechanism.
- *ConceptBase*: ConceptBase is a meta-modelling tool that is designed in a very generic way, which provides possibilities for the specification of modelling languages particularly with the aid of an enclosed programming environment. As a result of the expressive power of the programming language, nearly any model modification can be realised with ConceptBase. However, the evaluation showed that the low user-friendliness of ConceptBase makes the practical use nearly impossible.
- *Cubetto*: The evaluation result of Cubetto is similar to that of ConceptBase. Just as ConceptBase, Cubetto is not configuration-oriented. Nevertheless, nearly all requirements of information model variant management can be satisfied by Cubetto, since the corresponding mechanisms are programmed by hand. A script language with an appropriate expressive power is built-in. In comparison to ConceptBase, Cubetto offers a slightly enhanced user-friendliness.
- *GenGraph*: The meta modelling tool GenGraph is in the prototypical development stage and therefore only provides rudimentary functionalities. E.g., model type selection and element type selection are implemented like in the ARIS Toolset.
- *Metis*: In Metis it is possible to realise the variation of symbols. Furthermore, label switching tables can be defined allowing synonym management. Other variant building forms, possibilities of specification and query of perspectives are very difficult if not impossible to realise.
- *SemTalk*: SemTalk is – like indapta – a modelling plug-in for Microsoft Visio. The plug-in does basically not support model variants. According to the documentation of SemTalk the specification of Visual Basic macros is possible. However, its functionalities are not documented, so that we cannot judge if the script language is suited for the specification of configuration mechanisms.

Due to the missing variant building support of most of the examined modelling tools as well as the low practicability of tools that provide programming environment for the specification of configuration mechanisms, we decided to develop the new tool *indapta*. The goal was to provide a modelling tool

- that complies with the requirements of modellers from practice and research,
- that consequently provides a comprehensive configuration support, and
- that provides an easy-to-use modelling and variant building environment.

7 CONCLUSION AND OUTLOOK

Our design science research process was divided into three phases: identifying relevant configuration mechanisms through empirical work; conceptualising these mechanisms; and evaluating these concepts by showing their feasibility by means of a software prototype. Further research will extend the evaluation of the tool within different approaches. The tool will be applied to develop comprehensive multi-perspective reference information models for different domains. This project will help to evaluate the usability of the tool depending on ambitious model extents. In our reference model projects, modelling experts will be primarily involved. Additional evaluation approaches must address users which are less familiar with our concept of multi-perspective modelling and for this reason we will investigate the use of our tool in teaching and enterprise practices. The advantage of an academic context is that the experiments can be executed with a lot of comparable students. Giving them standardised assignments we can gather quantitative and qualitative data, which shows prosperities and failures appearing by using the tool. Naturally, the observation of the tool application in enterprise practice is less controllable but more realistic, therefore motivating the investigation of the tool support by case studies of projects, in which the design of different aspects of information systems demands multi-perspective modelling. One research objective of the case studies will be to check the expectance that the collaborative use of multi-perspective models assists the coordination of problem solving.

References

Becker, J., Algermissen, L., Delfmann, P. and Niehaves, B. (2006a). Configurable Reference Process Models for Public Administrations. In Encyclopedia of Digital Government. Editors: Anttiroiki, A.-V. and Mälkiä, M. Hershey et al., pp. 220-223.

Becker, J., Delfmann and Knackstedt (2007). Adaptive Reference Modeling: Integrating Configurative and Generic Adaptation Techniques for Information Models. In Proceedings of the Reference Modeling Conference 2006. Passau.

Becker, J., Janiesch, C., Delfmann, P. and Fuhr, W. (2006b) Perspectives on Process Documentation – A Case Study. In Enterprise Information Systems VII. Editors: Chen, C.-S., Filipe, J., Seruca, I. and Cordeiro, J. Dordrecht, pp. 167-177.

Becker, J., Kugeler, M. and Rosemann, M. (2003). Process Management. A Guide for the Design of Business Processes. Berlin et al.

Chen, P.P.-S. (1976) The Entity-Relationship Model: Toward a Unified View of Data. ACM Transactions on Database Systems 1 (1), pp. 9-36.

Darke, P. and Shanks, G. (1996). Stakeholder Viewpoints in Requirements Definition. Requirements Engineering 1 (1), pp. 88-105.

Delfman and Knackstedt (2007). Konfiguration von Informationsmodellen – Untersuchungen zu Bedarf und Werkzeugunterstützung. In eOrganisation: Service-, Prozess-, Market-Engineering. Volume 2. Editors: Oberweis, A., Weinhardt, C., Gimpel, H., Koschmider, A., Pankratius, V. and Schnizler, B. Karlsruhe. pp. 127-144.

ESPRIT Consortium AMICE (1989). CIM-OSA. Open System Architecture for CIM, Berlin et. al.

Finkelstein, A., Kramer, J., Nuseibeh, B., Finkelstein, L. and Goedicke, M. (1992). Viewpoints: a framework for integrating multiple perspectives in system development. International Journal of Software Engineering and Knowledge Engineering 2 (1), pp. 31-57.

Frank, U. (1999). Memo – Visual Languages for Enterprise Modelling. Working Papers of the Department for Information Systems, Koblenz.

Hevner, A.R., March, S.T., Park, J. and Ram, S. (2004). Design Science in Information Systems Research. MIS Quarterly, 28 (1), pp. 75-105.

Karimi, J. (1988). Strategic Planning for Information Systems: Requirements and Information Engineering Methods. Journal of Management Information Systems 4 (4), pp. 5-24.

Kottemann, J.E., Konsynski, B.R. (1984). Information Systems Planning and Development: Strategic Postures and Methodologies. Journal of Management Information Systems 1 (2), pp. 45-63.

Luhmann, N. (1999). Zweckbegriff und Systemrationalität. Über die Funktion von Zwecken in sozialen Systemen. 6th Edition. Frankfurt am Main.

Newell, A. and Simon, H.A. (1972). Human problem solving. 2nd Edition. Englewood Cliffs.

Rosemann, M. (1998). Managing the Complexity of Multi-perspective Information Models using the Guidelines of Modeling. In Proceedings of the 3rd Australian Conference on Requirements Engineering, pp. 101-118. Geelong.

Rosemann, M. and van der Aalst, W.M. P. (2007) A Configurable Reference Modelling Language. Information Systems 23 (1), pp. 1-23.

Scheer, A.-W. (2000). ARIS – Business Process Modelling. 3rd Edition. Berlin.

Soffer, P., Golany, B. and Dori, D. (2003) ERP modeling: a comprehensive approach. Information Systems 28 (9), pp. 673-690.

Zachman, J.A. (1987). A Framework for Information Systems Architecture. IBM Systems Journal 26 (3), pp. 277-293.